

EVOLVING OPTIMIZED VIDEO PROCESSING AND WIRELESS TRANSMISSION SYSTEM BASED ON ARM-CORTEX-A8 AND GSM

G. RAMENDER¹, MOVVA PAVAN² & K. KISHORE KUMAR³

¹Department of ECE, Aurora's Technological & Research Institute, Hyderabad, Andhra Pradesh, India

²Associate Professor, Department of ECE, Aurora's Technological & Research Institute, Hyderabad, Andhra Pradesh, India

³Assistant Professor & HOD, ECE, Faculty of Science & Technology, IFHE University, Hyderabad,
Andhra Pradesh, India

ABSTRACT

In this paper an optimized video processing concept is implemented as an application to surveillance monitoring system (SMS), mainly for maintaining an efficient database for SMS running on Samsung's S5PV210-CORTEX-A8 embedded application processor, expanding peripheral devices using embedded Linux as the operating system. Here the UVC driver and V4L programming is used to interface USB camera to the board to capture video information and then the board will do two parallel works, first one will transmit the processed captured video information using wireless network, second one will extract the key frames using Chi-Square Histogram difference algorithm with OpenCV tool, which greatly supports Image Processing applications on embedded devices and then served on a server. In general current SMS are either webcam based or simple motion detection based, webcam based system captures images and stores them even when the intrusion does not happen it leads to difficulty for the observer to locate the frame of interest also leads to excessive use of storage where as motion detection system does not provide the evidence of the intrusion. So, we have interfaced both for enough intelligence, mainly the interface provided by cortex-a8 for PIR sensor based motion detector and GSM communication module are additional things so that when motion sensor detects human entry in the remote security area the client will get an SMS at the same time by using a "smart device" the user can link to the video streaming server from a web browser constructed with in the embedded board via the internet and browse the webpage to monitor the remote location, also the client can analyses the key frames which are being served on web server and can take necessary actions. The entire system is running with Ubuntu-12.04 LTS Linux distribution and kernel-2.6.35 patch over Samsung S5PV210 Cortex-A8 processor as the processing unit.

KEYWORDS: Chi-Square Histogram Difference, Key Frame Extraction, Open CV Tool, Samsung S5PV210 Cortex-A8 Processor, Video Capture, Video Compression, Video Streaming Server

INTRODUCTION

In the current information era, information are represented and processed in the forms of multimedia. Especially in video processing, numerous frames containing similar information are usually processed.

This leads to unnecessary time consumptions, slow processing speed and complexity. Video summarization using key frames can facilitate to speed up video processing and so key frame extraction have become crucial for the development of advanced digital video systems also the Internet has very popular, and many kinds of consumer electronic products posses communication ability. Therefore, connecting a surveillance system to the "smart devices" can be an important application. Both the platform and the communication technology of the modern surveillance system have been improved to the point of achieving a high level of video streaming and real time image processing.

Many engineers design a smart camera on the embedded system and use image processing methods to analyze and judge surrounding things in home surveillance or traffic surveillance. Although image processing methods achieve the function of surveillance, they have to be implemented by choosing a PC with high speed and real time processing and for this reason a design using a PC cannot develop the advantages of low power consumption and low cost besides, we therefore use an embedded system instead of PC as the platform to implement a surveillance system with its advantage of low power consumption, small volume, low cost and high mobility for to handle the image data packets and compress the packets for the wireless access point server so that by utilizing “smart devices” the user can access the immediate video via the Internet.

In view of existing designs and previous research projects, we focus on selecting an adequate embedded board which can support video streaming server [1][6] and a webpage browsing function for key frames from captured video [6] which are being served on server.

Organization of the work is as follows: In Section II, we introduce the design of the system hardware and software. In section III, we introduce the key frame extraction and OpenCV. In section IV, we show the measurement result of the surveillance system and key frame extraction and in section V, we draw the conclusion.

HARDWARE AND SOFTWARE DESIGN OF THE SYSTEM

Hardware Architecture

Figure 1 shows the Hardware architecture and internal process steps of intelligent surveillance monitoring system. The hardware system includes processor, video-capture device and router to receive video information through Wi-Fi, PIR sensor and GSM module for sending an SMS to client.

Samsung S5PV210 Processor

The mini210 development board is a powerful Cortex-A8 board offering a comprehensive solution integrating both hardware and software. It is designed, developed and distributed by friendly ARM. It uses Samsung's S5PV210 microprocessor [7] whose maximum frequency is up to 1GHz. The Cortex-A8 high-performance processor is proven in end devices today. From high-end feature phones to netbooks, DTVs, printers and automotive-infotainment, the Cortex-A8 processor offers a proven high-performance solution with millions of units shipped annually.

The Cortex-A8 is designed to meet the needs of markets requiring high performance with power-efficiency, often integrating web connectivity. The S5PV210 integrates the powerVR SGX540 graphic engine, supports 3D and can drive video playing on screens up to 1080P.

The Mini210 inherits all the features and benefits of our popular Mini2440 and Mini6410 excelling in quality and easy to use with low cost. It is equipped with a 5" LCD, 512M DDR2, 1G SLC NAND flash, SD Wi-Fi, D type WM8960 audio which supports 8Ω 1W speakers. In addition it has a mini HDMI output, USB2.0 camera and 8x8 matrix key board.

It also supports power idle mode. These features make it easily and widely used in MID development, Android notepads, auto electronic devices, industrial applications, GPS systems and multimedia systems. It is very easy and convenient for users to refresh the system with various OS via a TF card with our specially developed super boot.

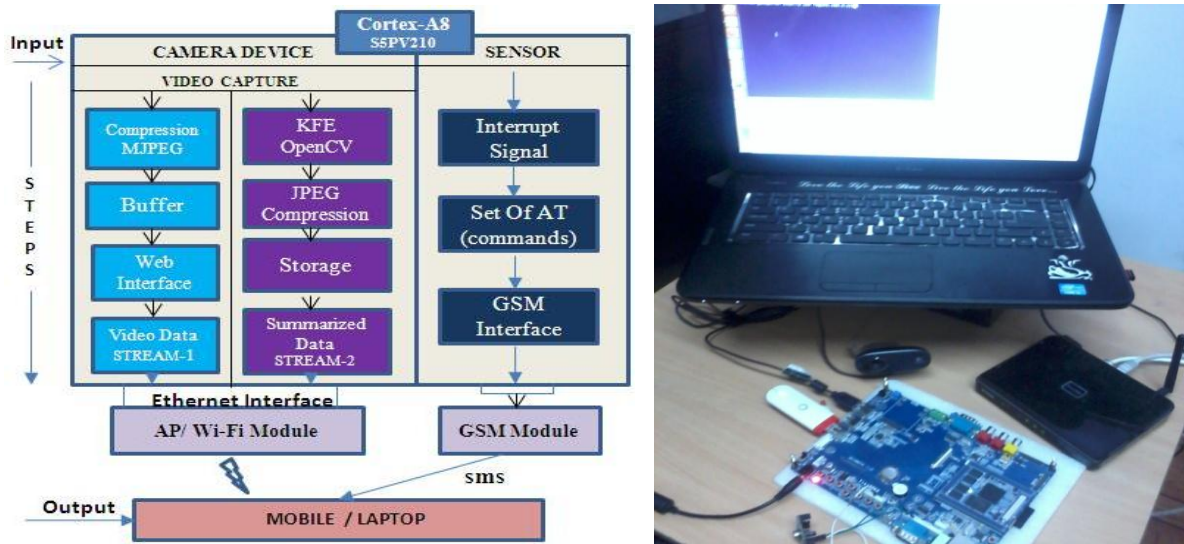


Figure 1: Hardware Architecture and Internal Process Steps of Intelligent Surveillance Monitoring System and Snapshot Showing Worked Hardware Architecture

The internal process flow of the system is shown in Figure 1 and the process is mentioned below as steps:

Step 1: When a person passes near to PIR sensor then it triggered and send alert message to user. After seeing that message the user can link to the video streaming server 1 or server 2 constructed on the embedded board via the internet and then browse the webpage to monitor the surveillance area.

Step 2: The camera continually monitor the room and captures the video, this captured video is compressed using MJPEG compression [9][11-12] technique and display in the server 1. Compression of video must be done, because some mobile devices allow only restricted amount of packet size between the server and themselves.

Step 3: From the captured video the key frames are extracted using OpenCV and compressed as JPEG [5][10] and saved image in SD card and also served on server 2, one advantage of this is whenever motion is detected and as soon we connected to server there may not be any evidence so key frames which already being stored provides the evidence.

Web Camera

Logitech USB2.0 camera is selected in the built system. This web camera continuously monitors the room and sends the video. Here UVC driver acts as an interface module which is being controlled by V4L2 interface [8]. The configurations can be done to camera as needed.

Pyroelectric Infrared (PIR) Sensor and GSM/3G Module

In this paper the PIR sensor is used to sense whether someone is passing through the surveillance area or not. The PIR sensor [15] receives the variations of the temperature made by someone emitting infrared energy to the surroundings it produces the variations of electric charges by means of a Pyroelectric effect. Because the electric charges are very few and not easily sensed, we adopt the high impedance FET to pick up the signal. In addition there is a disadvantage in the output response of the sensor. Since the output voltage we measure, about the level of mV, is too small, we have to amplify the out signal from the sensor with two-stage high-gain amplifiers. Nevertheless, if the gain is very high, tiny noises are amplified simultaneously and interfere seriously with the output signal. In our design we use the GPIO driver interface to provide the communication between the PIR sensor and the embedded board, if an intruder enters the surveillance area, the sensor is triggered and the system sends SMS to user mobile phone as “motion detected” using GSM module with help of attention commands which are mentioned in code.

Router

Router is a device that forwards the data packets between computer networks. Router is connected to Cortex-a8 board through Ethernet cable. Using Wi-Fi we can stream the video and key frames through mobile or laptop, the captured video data is stored in to the embedded web server using web interface after being processed. In the video streaming server the video transmission packets are transmitted via TCP/IP protocol, the server streams the sequence of JPEG frames over HTTP. A special mime-type content type multipart/x-mixed-replace; boundary=boundary-name informs the client to expect several parts (frames) as an answer delimited by boundary-name. This boundary name is expressly disclosed within the MIME-type declaration itself. The TCP connection is not closed as long as the client wants to receive new frames and the server wants to provide new frames. Here the router is configured to particular IP address and port and at the client side using HTTP protocol links to the server and browse the video stream data as well as key frames which are also being served on web server.

SOFTWARE ARCHITECTURE

Figure 2 shows the development procedure of the embedded software design. In cooperation with the operating system of the embedded system, we develop the application program and the debugging program at the PC host. By using a cross compiler we compile the embedded application and bundle them into the specific OS that is suitable for the hardware platform.

Figure 3 shows the software architecture of the system in which we adopt Embedded Linux as the system OS [2]. Embedded Linux is for embedded boards which can be used in mobile devices, home appliances, embedded devices and single purpose systems. This OS not only excels at network communication, but also conforms to GPL and features of free Software

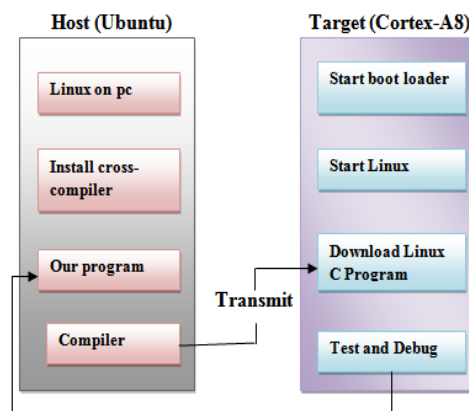


Figure 2: Development Procedure of the Video Monitoring SYSTEM

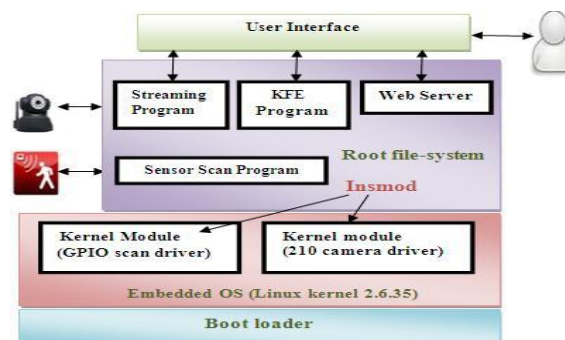


Figure 3: Software Architecture of the Embedded Software Design

Physically the architecture of embedded Linux and Linux are very similar. However, Embedded Linux is a kernel specified on a specific platform and for specific sets of libraries and utilities. Thus, the format of Embedded Linux is a combination of a kernel image and a root-file system image. It can be built on different kinds of embedded platforms, and this OS can be tailored to offer specific functions to specific users. We use the Linux c language for programming. Linux includes a series of compiler tools and debugging tools as well as a function toolbox. Moreover, it not only features high-level language, but also low-level language for easy control of the hardware. Linux C is therefore an adequate language for us to develop a program in an embedded system.

Our software program consists of 4 parts as shown in Figure 3:

- **Streaming Program:** Captures dynamic video
- **Key Frame Extraction Program:** Captures dynamic images
- **Sensor Scan:** Scans trigger from GPIO
- **Web Server:** Builds a webpage on embedded system

Figure 3 shows the kernel module which includes a GPIO scan driver and a camera module driver. When we recompile the Linux kernel, we have to combine them as a module. The camera can be used on an Embedded Linux OS. On the Linux platform the drivers are the subsets in the kernel module, and they are compiled with the kernel by the gcc compiler, but the drivers cannot be combined with the kernel as an image file. Only when we launch the camera will the drivers be downloaded (insmod) to the kernel. This method reduces the memory consumption and prevents the OS from not being able to be booted normally when the drivers fail.

KEYFRAME EXTRACTION AND OPENCV

Key frame extraction algorithm is implemented on embedded application processor Cortex-A8 board using OpenCV tool where the captured video is processed from which key frames are extracted based on histogram difference algorithm. The existing algorithms are well processed under matlab which creates a problem and does not meet requirements as for embedded devices concerned, so we selected OpenCV tool which greatly supports Image Processing applications[13] on embedded devices for implementing this design, also here the application is designed on board so that as per speed is concerned the histogram difference algorithm[3-4] is developed for extracting key frames followed with jpeg compression and finally the receiver on connecting to the server observes those extracted key frames and do necessary actions.

Steps

A color histogram is a representation of the distribution of colors in image.

Represent histogram by taking BIN value.

The value stored in each bin is the number of pixels in the image that shows the range

These ranges represent different levels of intensity of each RGB component.

- Collect the frames from the input video
- The conversion of RGB to HSV is done.
- Compute histogram for each channel(HSV)

- Normalize the histogram
- Compare the Histogram difference between the previous and the current frame using chi-square method
- $D(H_t, H_{t+1}) = \sum_{k=1}^N (H_t^k - H_{t+1}^k)^2 / (H_t + H_{t+1})$ Where H_t Denotes image histogram for the t^{th} frame, H_t^k denotes the values on the
- k^{th} colour H_t .
- The calculated sum is compared with empirically found threshold.
- If the sum greater than threshold then we obtain the key frames and then served on web server.

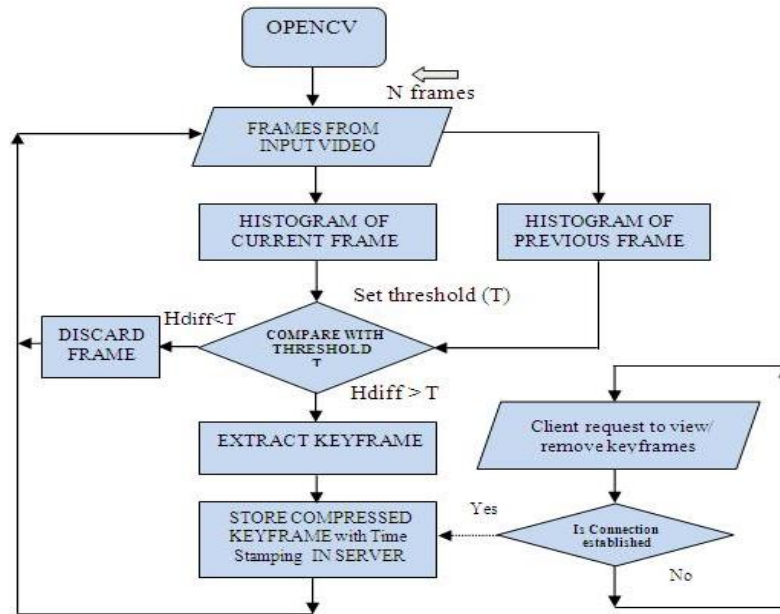


Figure 4: Flow Chart of Key Frame Extraction

Installing Configuring and Cross-Compiling OpenCV

Video surveillance applications provide another perspective on the evolution of embedded vision. Traditional surveillance systems are less concerned with vision analytics than they are with simply encoding and recording video data. However, as vision algorithms improve, video surveillance will incorporate more automated monitoring and analysis of this recorded data.

OpenCV is a collection of software algorithms put together in a library that to be used by industry and academics for computer vision applications and research, Using this libraries we design applications. To build OpenCV [13-14] we need to install some dependency libraries, Using below command we can install those libraries

```
$sudo apt-get install libjpeg-dev libjasper-dev libtiff4-dev libpng12-dev libpango1.0-dev libcairo2-dev libgdk-pixbuf2.0-dev libgtk2.0-dev cmake cmake-gui
```

After installing those libraries download the latest versions of ffmpeg x264 OpenCV source from git-repository and install them, to configure and install the OPENCV use below commands:

```
$cmake-gui $make && make install
```

Cross-Compile OpenCV and its libraries, to run on the board.

Steps:

A filesystem and a tool chain with the OpenCV libraries for Cortex A8 platform can be generated using the Open embedded tool Narcissus.

- Set up the compiler tool chain.
- Build v4l-utils
- Configure OpenCV
- Build and install OpenCV

Finally the generated libraries are ported along with the root filesystem, for to support executing OpenCV applications.

EXPERIMENTAL RESULTS

Web camera captures the video and allows the user to monitor by video streaming. We implement the software modules in the Linux C language for the video display and for the triggered signal while sensing the temperature. We also build a streaming server and a web server on the embedded board to allow the user to browse the surveillance system. In streaming server some linux web-streaming packages to be installed and configured on board (streaming server) I used ffmpeg, in which mjpeg as the video codec, vlc and apache2 etcetra to start streaming and it works ok, this implements the embedded server using TCP/IP protocol and is streamed at the client side using HTTP protocol [1][6].



Figure 5: Implementation of Displaying a Real Time Live Video Streaming on the Web Server

Experimental Setup of Design

At first, the parameters related to the video sequences are considered as illustrated in table 1 based on ARM embedded platform the proposed system process a video frame measuring 352*288(CIF) pixels at an interval of 40ms per frame then based on setup for an amount of time interval the obtained key frames from above mentioned steps are noted at different thresholds as illustrated in table 2. As can be seen in the results here the key frames are changing according to threshold value, the higher the threshold the lesser the key frames and lower the threshold higher the key frames so based on application we can make a relevant threshold so that we can save the transmission bandwidth accordingly.

Key Frame Results

Table 1: Experimental Setup

| S.No | Parameters | Specifications |
|------|---------------------------|-------------------------------|
| 1 | Video length (assumption) | 60 sec |
| 2 | Frame rate | 25fps |
| 3 | Resolution | 352x288=101376 pixels/frame |
| 4 | Number of Frames | 60x25=1500 |
| 5 | Bit rate (compressed) | 60x512kbps=30720kbps = 30Mbps |

Table 2: Result Analysis

| S.No | Threshold (k) | Actual Key Frames | Detected Key Frames | Detection Rate of Key Frames |
|------|---------------|-------------------|---------------------|------------------------------|
| 1 | K=50 | 30 | 06 | 20.00% |
| 2 | K=20 | 30 | 14 | 46.60% |
| 3 | K=15 | 30 | 19 | 63.33% |
| 4 | K=12 | 30 | 24 | 80.00% |
| 5 | K=10 | 30 | 29 | 96.66% |
| 6 | K=05 | 30 | 42 | N/A |

The relevant threshold-10 from above table 2 is applied for different videos and found that the detection rate of key frames extracted are satisfactory as shown in table 3.

Table 3: Quantitative Experimental Data of Proposed System for Key Frame Extraction Detection Accuracy

| Test Videos | No. of Actual Key Frames | No. of Correctly Determined Key Frames | Detection Accuracy |
|--|--------------------------|--|--------------------|
| Test video 1 | 70 | 68 | 97.14% |
| Test video 2 | 57 | 54 | 94.73% |
| Test video 3 | 30 | 29 | 96.66% |
| Total no. of actual key frames | 157 | | |
| No. of correctly determined key frames | 151 | | |
| Average key frame detection accuracy | 96.17% | | |

The below shown Figure 6 represents the key frames which are extracted from live video using CORTEX-A8 board and served on web server and Figure 7 shows the final output key frames with time stamping.



Figure 6: Snapshot Showing the Output of Key Frames



Figure 7: Snapshot Showing Output Key Frames for above Links Shown in Mobile

For full footage the user can link to live stream page on laptop or mobile as shown in Figure 5 also the obtained key frames are displayed along with time stamping so that the user can browse the images as soon as he gets an alert message and can open multiple tabs in mobile as shown in Figure 6.

CONCLUSIONS

In this paper the design and implementation of the moving object detection using open source computer vision and remote video monitoring system based on Cortex-A8 board is proposed. Based on an integrated design, we have combined PIR sensor circuit to detect the temperature changes of the surveillance area as a trigger to send SMS to the user, the Logitech Camera module to capture the video and video streaming server on the embedded board to view the captured video, with extraction of the key frame from the captured video on the Cortex-A8 board we save the network bandwidth accordingly and memory as well for embedded devices concerned. We have adopted Embedded Linux as the operating system on the embedded board because Linux supports well on the network and many free required modules can be selected.

REFERENCES

1. Yoshiro Imai, Yukio Hori "A Mobile Phone-Enhanced Remote Surveillance System with Electric Power Appliance Control and Network Camera Homing, IEEE Computer (Third International Conference on Autonomic and Autonomous Systems ICAS 2007).
2. Shao Guojin, Shen Yunqin "Embedded Linux Core Transplantation and Development of Handheld Terminals System" IEEE, Volume 5, pp 583- 586 978-1-4244-5586-October 2010.
3. Huayong Liu, Wenting Meng, Zhi Liu "Key Frame Extraction of Online Video Based on Optimized Frame Difference", 9th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2012)
4. Sagar Badnerkar, Prashant Ingole "Motion Sensed Video Storage Algorithm for Surveillance Recording", 12th international Conference on Hybrid Intelligent Systems (HIS) 2012.
5. CAPON, J.(1959). A Probabilistic Model for Run-length Coding of Pictures. IRE Trans on Information Theory, IT-5(4), pp.157-163.
6. G.Ramender and M.Pavani "Embedded Real Time Video Monitoring System Using ARM11 and GSM " International Journal of Electrical and Electronics Engineering Research (IJEEER), Volume (3)-Issue (4) pp 179-186, Oct 2013.
7. S5PV210 Microcontroller, <http://www.friendlyarm.net/products/tiny210> User's Manual.
8. Video4Linux User's Manual, <http://v4l.videotechnology.com/>
9. The Moving Picture Experts Group home page. <http://www.chiariglione.org/mpeg>
10. MPEG and JPEG standards <http://www.seas.gwu.edu/~ayoussef/cs225/standards.html>
11. IP-Camera MJPEG HTTP Web-Server Emulator <http://ipcamemu.codeplex.com/>
12. MJPEG:<http://www.digitalpreservation.gov/formats/fdd/fdd000063.shtml>
13. <http://docs.opencv.org/>
14. <http://ramenderg.wordpress.com/>
15. [15] PIR Introduction: <http://designer.mech.yzu.edu.tw/>

